UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/717,675	11/21/2000	Christopher G. Kaler	777.340US1	9107

41505 7590 05/16/2005

WOODCOCK WASHBURN LLP
ONE LIBERTY PLACE - 46TH FLOOR
PHILADELPHIA, PA 19103

EXAMINER	
ALI, SYED J	

ART UNIT	PAPER NUMBER
2195	

DATE MAILED: 05/16/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/717,675	KALER ET AL.	
	Examiner	Art Unit	
	Syed J. Ali	2195	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *17 February 2005*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-71* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-71* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This office action is in response to the amendment filed February 17, 2005. Claims 1-71 are presented for examination.

2. The text of those sections of Title 35, U.S. code not included in this office action can be found in a prior office action.

Claim Rejections - 35 USC § 102

3. **Claims 1-10, 38-40, 49-52, 55-56, 58-59, and 65-66 are rejected under 35 U.S.C. 102(e) as being anticipated by Shah et al. (USPN 6,226,689) (hereinafter Shah).**

4. As per claim 1, Shah teaches the invention substantially as claimed, including a method in a computer system for servicing requests from one or more client computers, the method comprising:

receiving a request from a client computer (col. 3 lines 51-54);

a first thread processing the request by invoking a receive handler that creates a work item, wherein the first thread is part of a pool of generic threads (col. 3 lines 51-54);

a second thread performing a task specified in the work item by invoking a work handler, wherein the second thread is part of the pool of generic threads (col. 4 lines 58-67);

receiving a result of performing the task (col. 3 lines 54-57); and

a third thread returning at least a portion of the result to the client computer by invoking a reply handler, wherein the third thread is part of the pool of generic threads (col. 3 lines 54-57).

5. As per claim 2, Shah teaches the invention as claimed, including the method as claimed in claim 1, wherein receiving the request comprises:

receiving the request into an input/output port (col. 6 lines 36-41); and

placing a reference in a queue indicating that work is available for the first thread (col. 6 lines 59-65).

6. As per claim 3, Shah teaches the invention as claimed, including the method as claimed in claim 2, further comprising:

invoking a receive thread manager when the work is available (col. 6 lines 46-50); and

the receive thread manager scheduling the first thread for execution on one of multiple processors (col. 4 lines 58-67).

7. As per claim 4, Shah teaches the invention as claimed, including the method as claimed in claim 3, wherein the receive thread manager schedules the first thread from a queue of available threads (col. 6 lines 46-67).

8. As per claim 5, Shah teaches the invention as claimed, including the method as claimed in claim 1, further comprising the first thread placing the work item on a work queue for execution by the second thread (col. 6 lines 59-65).

9. As per claim 6, Shah teaches the invention as claimed, including the method as claimed in claim 5, further comprising the first thread placing a reference in a completion port queue, indicating that work is available for the second thread (col. 6 lines 36-41).

10. As per claim 7, Shah teaches the invention as claimed, including the method as claimed in claim 1, further comprising the second thread creating and placing a second work item on a reply work queue when the results are received (col. 7 lines 17-27).

11. As per claim 8, Shah teaches the invention as claimed, including the method as claimed in claim 7, further comprising the second thread placing a reference in a completion port queue, indicating that work is available for the third thread (col. 6 lines 36-41; col. 7 lines 17-27).

12. As per claim 9, Shah teaches the invention as claimed, including the method as claimed in claim 1, further comprising:

receiving input data (col. 6 lines 1-45);

the first thread storing the input data in a cache that is accessible to the second thread (col. 6 lines 1-45).

13. As per claim 10, Shah teaches the invention as claimed, including the method as claimed in claim 1, wherein the results include data, and further comprising storing the data in a cache that is accessible to the third thread (col. 6 lines 1-45).

14. As per claim 38, Shah teaches the invention as claimed, including a method in a computer system for servicing requests from one or more client computers, the method comprising:

maintaining a pool of threads, wherein each thread in the pool of threads is identical and can invoke at least one receive handler and at least one work handler (col. 3 lines 57-61);

invoking receive handlers by the threads in response to receiving requests from one or more client computers (col. 3 lines 51-54), wherein the receive handlers create work items that specify tasks to be performed to satisfy the request (col. 4 lines 58-67);

invoking work handlers by the threads to perform the tasks specified in the work items (col. 4 lines 58-67);

receiving results of the tasks (col. 3 lines 54-57); and

invoking reply handlers by the threads to return at least portions of the results to the client computers (col. 3 lines 54-57).

15. As per claim 39, Shah teaches the invention as claimed, including the method as claimed in claim 38, wherein one or more of the receive handlers, one or more of the work handlers, and one or more of the reply handlers can be simultaneously executed on multiple processors available to the computer system (col. 3 lines 18-24; col. 3 lines 48-61).

16. As per claim 40, Shah teaches the invention as claimed, including the method as claimed in claim 38, wherein the pool of threads includes a number of threads in a range from about N+1

to about 2*N, where N is a number of processors available to the computer system (col. 3 lines 48-61).

17. As per claim 49, Shah teaches the invention as claimed, including an application program for implementation by an application server, the application program comprising:

at least one receive handler that can be invoked by a thread within a pool of threads (col. 3 lines 51-54), wherein each thread in the pool of threads is identical (col. 3 lines 57-61);

at least one work handler that also can be invoked by the thread (col. 4 lines 58-67); and

at least one reply handler that also can be invoked by the thread (col. 3 lines 54-57).

18. As per claim 50, Shah teaches the invention as claimed, including the application program as claimed in claim 49, wherein a receive handler of the at least one receiver handler is executed when the application server receives a request from a client computer (col. 3 lines 54-57), and the receiver handler creates a first work item to be performed by a work handler of the at least one work handler (col. 3 lines 51-54).

19. As per claim 51, Shah teaches the invention as claimed, including the application program as claimed in claim 50, wherein the work handler is executed when the first work item exists (col. 6 lines 46-50), the work handler receives results (col. 3 lines 54-57), and the work handler creates a second work item to be performed by a reply handler of the at least one reply handler (col. 6 lines 36-41; col. 7 lines 17-27).

20. As per claim 52, Shah teaches the invention as claimed, including the application program as claimed in claim 51, wherein the reply handler is executed when the second work item exists, and the reply handler sends the results to the client computer (col. 3 lines 54-57; col. 6 lines 36-41; col. 7 lines 17-27).

21. As per claim 55, Shah teaches the invention as claimed, including the application program as claimed in claim 49, wherein multiple identical copies of the thread exist within the pool of threads, and at least one copy of the thread is executed each time a request is received from a client computer (col. 3 lines 48-61).

22. As per claim 56, Shah teaches the invention as claimed, including the application program as claimed in claim 49, wherein multiple copies of the thread can simultaneously be executed by multiple processors available to the application server (col. 3 lines 18-24; col. 3 lines 48-61).

23. As per claim 58, Shah teaches the invention as claimed, including a computer system for servicing requests from one or more client computers, the computer system comprising:

a memory containing an application server which maintains a pool of threads, wherein each thread in the pool of threads is identical (col. 3 lines 57-61) and can invoke at least one receive handler (col. 3 lines 51-54), at least one work handler (col. 4 lines 58-67), and at least one reply handler (col. 3 lines 51-54) and the application server further schedules threads in the

pool of threads for execution on multiple processors available to the computer system (col. 3 lines 18-24; col. 3 lines 48-61); and

the multiple processors for simultaneously executing multiple threads within the pool of threads (col. 3 lines 18-24; col. 3 lines 48-61).

24. As per claim 59, Shah teaches the invention as claimed, including the computer system as claimed in claim 58, wherein the application server further performs functions of:

receiving a request from a client computer (col. 3 lines 51-54);

a first thread processing the request by invoking a receive handler which creates a work item, wherein the first thread is part of the pool of threads (col. 3 lines 51-54); and

a second thread performing a task specified in the work item by invoking a work handler, wherein the second thread is part of the pool of threads (col. 4 lines 58-67);

receiving a result of performing the task (col. 3 lines 54-57); and

a third thread returning at least a portion of the result to the client computer by invoking a reply handler, wherein the third thread is part of the pool of threads (col. 3 lines 51-54).

25. As per claim 65, Shah teaches the invention as claimed, including a computer-readable medium holding computer executable instructions, the computer-readable medium for performing a method in a computer system, the method comprising:

maintaining a pool of threads, wherein each thread in the pool of threads is identical (col. 3 lines 57-61) and can invoke at least one receive handler (col. 3 lines 51-54), at least one work handler (col. 4 lines 58-67), and at least one reply handler (col. 3 lines 51-54), and the

application server further schedules threads in the pool of threads for execution on multiple processors available to the computer system (col. 3 lines 18-24; col. 3 lines 48-61);

simultaneously executing multiple threads within the pool of threads on multiple processors available to the computer system (col. 3 lines 18-24; col. 3 lines 48-61).

26. As per claim 66, Shah teaches the invention as claimed, including the computer-readable medium as claimed in claim 65, wherein the method further comprises:

receiving a request from a client computer (col. 3 lines 51-54);

a first thread processing the request by invoking a receive handler, which creates a work item, wherein the first thread is part of the pool of threads (col. 3 lines 51-54);

a second thread performing a task specified in the work item by invoking a work handler, wherein the second thread is part of the pool of threads (col. 4 lines 58-67);

receiving a result of performing the task (col. 3 lines 54-57); and

a third thread returning at least a portion of the result to the client computer by invoking a reply handler, wherein the third thread is part of the pool of threads (col. 3 lines 51-54).

27. **Claims 41-48 are rejected under 35 U.S.C. 102(b) as being anticipated by Banks et al. (USPN 5,901,334) (hereinafter Banks).**

28. As per claim 41, Banks teaches the invention as claimed, including a method in a computer system for servicing requests from multiple client computers, the method comprising:

monitoring a quantity of work being performed by the computer system (col. 5 lines 44-47);

determining whether the quantity has exceeded an upper limit (col. 5 lines 54-56); and

if the quantity has exceeded the upper limit but has not dropped below a lower limit, not accepting new requests into the computer system (col. 5 lines 54-56).

29. As per claim 42, Banks teaches the invention as claimed, including the method as claimed in claim 41, further comprising, if the quantity has exceeded the upper limit and has dropped below the lower limit, accepting the new requests into the computer system (col. 5 lines 54-56; col. 5 line 66 - col. 6 line 15).

30. As per claim 43, Banks teaches the invention as claimed, including the method as claimed in claim 41, further comprising continuing to monitor the quantity of work being performed by the computer system after the quantity has exceeded the upper limit (col. 5 lines 44-47).

31. As per claim 44, Banks teaches the invention as claimed, including the method as claimed in claim 41, wherein the quantity of work is indicated by a number of work items on one or more work queues (col. 5 lines 44-47), and determining whether the quantity has exceeded the upper limit comprises determining whether the number of work items has exceeded a predefined value (col. 6 lines 66-67).

32. As per claims 45-48, Banks teaches the invention as claimed, including a method in a computer system for servicing requests from multiple client computers, the method comprising:

monitoring an amount of time to return a result by the computer system (col. 5 lines 44-47; col. 6 lines 66-67);

determining whether the amount of time has exceeded an upper limit (col. 5 lines 54-56); and

if the amount of time has exceeded the upper limit but has not dropped below a lower limit, not processing new work items (col. 5 lines 54-56).

33. As per claim 46, Banks teaches the invention as claimed, including the method as claimed in claim 45, further comprising, if the amount of time has exceeded the upper limit and has dropped below the lower limit, accepting and processing the new work items (col. 5 lines 54-56; col. 5 line 66 - col. 6 line 15).

34. As per claim 47, Banks teaches the invention as claimed, including the method as claimed in claim 45, further comprising continuing to monitor the amount of time after the amount of time has exceeded the upper limit (col. 5 lines 44-47; col. 6 lines 66-67).

35. As per claim 48, Banks teaches the invention as claimed, including the method as claimed in claim 45, wherein the amount of time includes a time it takes to post the result to an output port (col. 5 lines 44-47), and determining whether the amount of time has exceeded the

upper limit comprises determining whether the time it takes to post the result has exceeded a predefined value (col. 6 lines 66-67).

Claim Rejections - 35 USC § 103

36. **Claims 11-12, 14-16, 18-21, 23-25, 53-54, 60, and 67 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shah in view of Fant (USPN 6,327,607).**

37. As per claim 11, Fant teaches the invention as claimed, including the method as claimed in claim 1, further comprising:

determining a size of the result (col. 4 lines 11-56); and

when the size of the result is not larger than a cutoff size, storing the result in the partial results cache (col. 4 lines 11-56).

38. It would have been obvious to one of ordinary skill in the art to combine Shah and Fant since the use of a partial cache to store incremental results ensures that the results are not lost due to overflow errors, while also providing the return threads with a data store that is much faster than reading from main memory. The benefits of using cache to store data are well known, and to further refine the cache usage to include a partial results cache provides a means of guaranteeing that the data being returned makes optimal use of the size limitations of the cache.

39. As per claim 12, Shah teaches the invention as claimed, including the method as claimed in claim 11, further comprising:

the third thread returning a portion of the result to the client computer (col. 3 lines 54-57); and

if a second request is received for an additional portion of the result that is stored in the partial results cache, a fourth thread creating a second work item by invoking another receive handler, wherein the second work item is processed by a fifth thread, which invokes another reply handler (col. 3 lines 48-61; col. 4 lines 52-67).

40. As per claim 14, Shah teaches the invention as claimed, including a method in a computer system for servicing requests from multiple client computers, the method comprising:

receiving a request from a client computer to perform a function (col. 3 lines 51-54);

performing a first task, by a first work handler invoked by a first thread in a ready state, wherein the first task is associated with a first state of the function (col. 3 lines 51-54), performing the first task includes issuing an asynchronous request for data (col. 9 lines 28-30);

placing the first thread back in the ready state (col. 6 lines 65-67);

receiving the data specified in the asynchronous request (col. 11 lines 34-37); and

performing a second task, by a second work handler invoked by a second thread in the ready state, wherein the second task is associated with a second state of the function, and the second task performs an operation on the data (col. 4 lines 58-67), wherein the first thread and the second thread are all identical generic threads within a pool of threads (col. 1 lines 14-21).

41. Fant teaches the invention as claimed, including the request being a multi-state function (col. 7 lines 12-33; col. 7 lines 51-63).

42. As per claim 15, Shah teaches the invention as claimed, including the method as claimed in claim 14, further comprising:

processing the request by a receive handler invoked by a third thread (col. 3 lines 54-57);

creating a work item that specifies the first task (col. 4 lines 58-67); and

placing the work item in a work queue that is accessible to the first thread (col. 6 lines 36-41).

43. As per claim 16, Shah teaches the invention as claimed, including the method as claimed in claim 15, wherein the first thread, the second thread, and the third thread are all identical generic threads within the pool of generic threads (col. 3 lines 57-61).

44. As per claim 18, Shah teaches the invention as claimed, including the method as claimed in claim 14, wherein the asynchronous request is a request that would otherwise cause the first thread to block (col. 9 lines 28-30).

45. As per claim 19, Shah teaches the invention as claimed, including the method as claimed in claim 14, wherein the asynchronous request is a request that would otherwise cause a blocking condition to occur (col. 9 lines 28-30).

46. As per claim 20, Shah teaches the invention as claimed, including the method as claimed in claim 14, wherein issuing the asynchronous request comprises issuing the asynchronous request to a database manager (col. 9 lines 28-30; col. 11 lines 35-38).

47. As per claim 21, Shah teaches the invention as claimed, including the method as claimed in claim 20, further comprising:

the database manager placing the asynchronous request on a pending queue (col. 9 lines 28-30; col. 11 lines 35-38; col. 6 lines 59-65); and

when the data is received, placing a work item associated with the second state on a work queue (col. 6 lines 59-65).

48. As per claim 23, Shah teaches the invention as claimed, including the method as claimed in claim 14, further comprising returning a result of the first task and the second task by invoking third thread, which in turn invokes a reply handler to return the result to the client computer (col. 3 lines 54-57).

49. As per claim 24, Shah teaches the invention as claimed, including the method as claimed in claim 23, wherein the first thread, the second thread, and the third thread are all identical generic threads within a pool of generic threads (col. 3 lines 57-61).

50. As per claim 25, Shah teaches the invention as claimed, including the method as claimed in claim 14, further comprising:

performing additional tasks associated with subsequent function states by additional work handlers invoked by one or more additional threads (col. 6 lines 65-67);

at least some of the additional work handlers issuing additional requests (col. 7 lines 17-27); and

placing threads associated with the at least some of the additional work handlers back in the ready state after issuing the additional requests (col. 6 lines 65-67); and

the requests being asynchronous requests (col. 9 lines 28-30; col. 11 lines 35-38).

51. As per claim 53, Shah teaches the invention as claimed, including the application program as claimed in claim 49, wherein the thread can invoke multiple work handlers (col. 4 lines 58-67); and

at least some of the multiple work handlers issue asynchronous requests for data (col. 9 lines 28-30).

52. Fant teaches the invention as claimed, including where some of the multiple work handlers are designed to perform tasks associated with various states of a multi-state function, wherein the thread is then placed back in a ready state to execute a subsequent work item (col. 7 lines 12-33; col. 7 lines 51-63).

53. As per claim 54, Shah teaches the invention as claimed, including the application program as claimed in claim 53, wherein when the data is returned, a second work handler is executed (col. 3 lines 54-57).

54. As per claim 60, Shah teaches the invention as claimed, including the computer system as claimed in claim 58, wherein the application server further performs functions of:

receiving a request from a client computer to perform function (col. 3 lines 51-54);

a first thread within the pool of threads performing a first task by invoking a first work handler (col. 4 lines 58-67) where performing the first task includes issuing an asynchronous request for data (col. 9 lines 28-30);

placing the first thread back in a ready state (col. 6 lines 65-67);

receiving the data specified in the asynchronous request (col. 9 lines 28-30); and

a second thread within the pool of threads performing a second task by invoking a second work handler, and the second task performs an operation on the data (col. 4 lines 58-67).

55. Fant teaches the invention as claimed, including the request being a multi-state function (col. 7 lines 12-33; col. 7 lines 51-63).

56. As per claim 67, Shah teaches the invention as claimed, including the computer-readable medium as claimed in claim 65, wherein the method further comprises:

receiving a request from a client computer to perform a function (col. 3 lines 51-54);

a first thread within the pool of threads performing a first task by invoking a first work handler (col. 4 lines 58-67), where performing the first task includes issuing an asynchronous request for data (col. 9 lines 28-30);

placing the first thread back in a ready state (col. 6 lines 65-67);

receiving the data specified in the asynchronous request (col. 9 lines 28-30); and

a second thread within the pool of threads performing a second task by invoking a second work handler, and the second task performs an operation on the data (col. 4 lines 58-67).

57. Fant teaches the invention as claimed, including the request being a multi-state function (col. 7 lines 12-33; col. 7 lines 51-63).

58. **Claim 13 is rejected under 35 U.S.C. 103(a) as being unpatentable over Shah in view of Ramakrishnan et al. (USPN 6,085,215) (hereinafter Ramakrishnan).**

59. As per claim 13, Shah teaches the invention as claimed, including the method as claimed in claim 1, further comprising:

the first thread, the second thread or the third thread indicating that the first thread, the second thread or the third thread has completed a work item (col. 6 lines 36-41).

60. Ramakrishnan teaches the invention as claimed, including if a quantum has not expired for the first thread, the second thread or the third thread, then the first thread, the second thread or the third thread being given an additional work item to perform without relinquishing the central processing unit upon which the first thread, the second thread or the third thread was running (col. 9 line 52 - col. 10 line 3).

61. It would have been obvious to one of ordinary skill in the art to combine Shah with Ramakrishnan since the controlling of the amount of time a thread is scheduled to run eliminates livelock and starvation conditions by guaranteeing that a thread services a minimum amount of work and yields after a specified quantum, respectively. Ramakrishnan provides a minimum execution time for a thread, which is specified in terms of work units, such that a thread executes for a minimum amount of time, thereby allowing a thread to perform its requisite task without being interrupted. Additionally, Ramakrishnan provides the benefit of a maximum execution

time, also specified in terms of work units, that ensures that one thread does not occupy too many system resources by guaranteeing that it yields after a specified period of time.

62. **Claims 17, 22, and 26-32 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shah in view of Fant in view of Wight et al. (USPN 6,219,353) (hereinafter Wight).**

63. As per claim 17, Wight teaches the invention as claimed, including the method as claimed in claim 15, wherein placing the work item in the work queue comprises placing the work item in a low priority work queue, the method further comprising:

placing a second work item that specifies the second task on a high priority work queue (col. 4 lines 43-59);

a work handler looking for a work item first on the high priority work queue (col. 4 line 60 - col. 5 line 3); and

if no work item exists on the high priority work queue, the work handler looking for the work item on the low priority queue (col. 4 line 60 - col. 5 line 3).

64. It would have been obvious to one of ordinary skill in the art to combine Shah, Fant, and Wight since the use of priority queues allows the system to ensure that important requests, such as requests with hard deadlines are serviced first. For instance, if a high priority important request is received, it may be detrimental to simply place the request in a first-in-first-out queue. A deadline may be missed or the system may lag. By providing separate queues for high priority and low priority requests, it can be ensured that the most important requests are serviced first.

65. As per claim 22, Wight teaches the invention as claimed, including the method as claimed in claim 21, wherein the work queue is a high priority work queue, the method further comprising:

placing a first work item that specifies the first task on a low priority work queue (col. 4 lines 43-59);

a work handler looking for a work item first on the high priority work queue (col. 4 line 60 - col. 5 line 3); and

if no work item exists on the high priority work queue, the work handler looking for the work item on the low priority work queue (col. 4 line 60 - col. 5 line 3).

66. As per claim 26, Shah teaches the invention as claimed, including a method in a computer system for servicing requests from multiple client computers, the method comprising:

determining that work is available after receiving a request from a client computer (col. 3 lines 54-57; col. 4 lines 58-67); and

when work is available, a first work handler invoked by a first thread looking in a first work queue for a first work item corresponding to the work (col. 4 lines 58-67), wherein the first thread is a generic thread within a pool of generic threads (col. 1 lines 14-21).

67. Fant teaches the invention as claimed, including the request from the client computer is a request to perform a function having multiple states (col. 7 lines 12-33; col. 7 lines 51-63).

68. Wight teaches the invention as claimed, including if the first work item is not found in the first work queue, the first work handler looking in a second work queue for the first work item (col. 4 line 60 - col. 5 line 3).

69. As per claim 27, Shah teaches the invention as claimed, including the method as claimed in claim 26, further comprising:

receiving a request from a client computer to perform a task (col. 3 lines 54-57);

creating the first work item that specifies the task (col. 3 lines 54-57; col. 4 lines 58-67);

placing the first work item in the second queue (col. 3 lines 54-57; col. 4 lines 58-67); and

indicating that the work is available (col. 4 lines 58-67).

70. As per claim 28, Shah teaches the invention as claimed, including the method as claimed in claim 26, further comprising:

the first work handler performing a task specified in the first work item (col. 4 lines 58-67); and

issuing an asynchronous request for data (col. 9 lines 28-30).

71. As per claim 29, Shah teaches the invention as claimed, including the method as claimed in claim 28, further comprising:

receiving the data (col. 6 lines 1-45);

placing a second work item on the first work queue (col. 4 lines 58-67; col. 6 lines 1-45); and

indicating that additional work is available (col. 4 lines 58-67; col. 6 lines 1-45).

72. As per claim 30, Wight teaches the invention as claimed, including the method as claimed in claim 29, further comprising:

a second work handler invoked by a second thread looking in the first work queue for the second work item when the second work is available (col. 4 line 60 - col. 5 line 3), wherein the first and second threads are all identical threads within the pool of generic threads (col. 1 lines 14-21); and

performing a second task specified in the second work item (col. 4 line 60 - col. 5 line 3).

73. As per claim 31, Wight teaches the invention as claimed, including the method as claimed in claim 26, wherein the computer system includes multiple work queues, including the first work queue and the second work queue, each of the multiple work queues are associated with a priority level, and wherein the method further comprises:

looking for the first work item first in a work queue associated with a highest priority level (col. 4 line 60 - col. 5 line 3); and

if the first work item is not found in the work queue associated with the highest priority level, looking for the first work item in each of the multiple work queues in descending priority order until the first work item is found (col. 4 line 60 - col. 5 line 3).

74. As per claim 32, Fant teaches the invention as claimed, including the method as claimed in claim 31, wherein each state of the multi-state function is performed by a work handler invoked by a subsequent thread based on work items placed in the multiple work queues (col. 7 lines 12-33; col. 7 lines 51-63).

75. Wight teaches the invention as claimed, including the work items are placed in higher and higher priority work queues as execution of the multi-state function progresses through the multiple states (col. 4 line 60 - col. 5 line 3).

76. **Claims 33, 36, 57, 62, and 69 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shah in view of Challenger et al. (USPN 6,026,413) (hereinafter Challenger).**

77. As per claim 33, Shah teaches the invention as claimed, including a method in a computer system for servicing requests from multiple client computers, the method comprising:

receiving, from a client computer, a request to perform a first task (col. 3 lines 51-54), wherein the task is handled by threads that are all identical threads within a pool of generic threads (col. 1 lines 14-21).

78. Challenger teaches the invention as claimed, including evaluating the first task, by a first handler invoked by a first thread, to determine whether the first task includes complex or long-running logic (col. 30 line 34 - col. 33 line 50); and

if the first task includes complex or long-running logic, performing the first task by a second handler invoked by a second thread (col. 30 line 34 - col. 33 line 50).

79. It would have been obvious to one of ordinary skill in the art to combine Shah and Challenger since a complex or long-running request may require special consideration. Specifically, a request that requires more processing than is required for a typical task may need to be serviced at a higher priority level since it may take longer to complete. Additionally, the

amount of time required to process the thread might be longer than the specified scheduling epoch, thereby requiring several time slices to complete the processing. A number of additional concerns may arise, requiring further attention. Therefore, it is important to identify these types of tasks and take the appropriate action.

80. As per claim 36, Shah teaches the invention as claimed, including the method as claimed in claim 33, further comprising:

the second handler returning a result of the first task (col. 3 lines 54-57); and

using the result, performing a second task by a third handler invoked by a third thread of the first group of threads (col. 3 lines 54-57).

81. As per claim 57, Challenger teaches the invention as claimed, including the application program as claimed in claim 49, further comprising one or more complex logic handlers that can be invoked by a second type of thread, wherein a thread of the second type is executed when a request from a client computer involves execution of complex or long-running logic (col. 30 line 34 - col. 33 line 50).

82. As per claim 62, Shah teaches the invention as claimed, including the computer system as claimed in claim 58, wherein the application server further performs functions of:

receiving, from a client computer, a request to perform a first task (col. 3 lines 51-54).

83. Challenger teaches the invention as claimed, including evaluating the first task, by a first handler invoked by a first thread within the pool of threads, to determine whether the first task includes complex or long-running logic (col. 30 line 34 - col. 33 line 50); and

if the first task includes complex or long-running logic, performing the first task by a second handler invoked by a second thread that is not within the pool of threads (col. 30 line 34 - col. 33 line 50).

84. As per claim 69, Shah teaches the invention as claimed, including the computer-readable medium as claimed in claim 65, wherein the method further comprises:

receiving, from a client computer, a request to perform a first task (col. 3 lines 51-54).

85. Challenger teaches the invention as claimed, including evaluating the first task, by a first handler invoked by a first thread within the pool of threads, to determine whether the first task includes complex or long-running logic (col. 30 line 34 - col. 33 line 50); and

if the first task includes complex or long-running logic, performing the first task by a second handler invoked by a second thread that is not within the pool of threads (col. 30 line 34 - col. 33 line 50).

86. **Claims 34-35 and 37 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shah in view of Challenger in view of Wight.**

87. As per claim 34, Wight teaches the invention as claimed, including the method as claimed in claim 33, wherein the first thread is a thread within a first group of threads having a

first priority level, and the second thread is a thread within a second group of threads having a second priority level that is lower than the first priority level (col. 4 line 43 - col. 5 line 3).

88. It would have been obvious to one of ordinary skill in the art to combine Shah, Challenger, and Wight since the use of priority queues allows the system to ensure that important requests, such as requests with hard deadlines are serviced first. For instance, if a high priority important request is received, it may be detrimental to simply place the request in a first-in-first-out queue. A deadline may be missed or the system may lag. By providing separate queues for high priority and low priority requests, it can be ensured that the most important requests are serviced first.

89. As per claim 35, Shah teaches the invention as claimed, including the method as claimed in claim 34, further comprising:

receiving a request to perform a second task (col. 3 lines 51-54).

90. Challenger teaches the invention as claimed, including evaluating the second task to determine whether the second task includes complex or long-running logic (col. 30 line 34 - col. 33 line 50); and

if the second task does not include complex or long-running logic and a processor is not available for performing the second task, preempting the second thread and performing the second task by a third thread in the first group of threads (col. 30 line 34 - col. 33 line 50).

91. As per claim 37, Wight teaches the invention as claimed, including the method as claimed in claim 36, wherein the first task is specified in a first work item and the second task is specified in a second work item, the method further comprising:

the first handler obtaining the first work item from a first work queue (col. 4 line 60 - col. 5 line 3); and

the third handler obtaining the second work item from a second work queue (col. 4 line 60 - col. 5 line 3).

92. **Claims 61 and 68 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shah in view of Wight.**

93. As per claim 61, Wight teaches the invention as claimed, including the computer system as claimed in claim 58, wherein the application server further performs functions of:

determining that work is available after receiving a request from a client computer (col. 4 line 60 - col. 5 line 3);

when work is available, a first thread within the pool of threads invoking a first work handler to look in a first work queue for a first work item corresponding to the work (col. 4 line 60 - col. 5 line 3); and

if the first work item is not found in the first work queue, the first work handler looking in a second work queue for the first work item (col. 4 line 60 - col. 5 line 3).

94. It would have been obvious to one of ordinary skill in the art to combine Shah and Wight since the use of priority queues allows the system to ensure that important requests, such as

requests with hard deadlines are serviced first. For instance, if a high priority important request is received, it may be detrimental to simply place the request in a first-in-first-out queue. A deadline may be missed or the system may lag. By providing separate queues for high priority and low priority requests, it can be ensured that the most important requests are serviced first.

95. As per claim 68, Wight teaches the invention as claimed, including the computer-readable medium as claimed in claim 65, wherein the method further comprises:

determining that work is available after receiving a request from a client computer (col. 4 line 60 - col. 5 line 3);

when work is available, a first work handler invoked by a first thread within the pool of threads looking in a first work queue for a first work item corresponding to the work (col. 4 line 60 - col. 5 line 3); and

if the first work item is not found in the first work queue, the first work handler looking in a second work queue for the first work item (col. 4 line 60 - col. 5 line 3).

96. **Claims 63-64 and 70-71 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shah in view of Banks.**

97. As per claim 63, Banks teaches the invention as claimed, including the computer system as claimed in claim 58, wherein the application server further performs functions of:

monitoring a quantity of work being performed by the computer system (col. 5 lines 44-47);

determining whether the quantity has exceeded an upper limit (col. 5 lines 54-56); and

if the quantity has exceeded the upper limit but has not dropped below a lower limit, not accepting new requests into the computer system (col. 5 lines 54-56).

98. It would have been obvious to one of ordinary skill in the art to combine Shah and Banks since there may be difficulty in identifying where a bottleneck is occurring since the receive, work, and reply handlers all reside on the same machine. Banks addresses this issue, stating that if the amount of time a request is in a queue is below an upper limit indicates that the work is being processed properly, and the bottleneck is occurring on the other machine. A series of steps are outlined that allow the system to refuse or reject additional incoming requests until the overload condition has been sufficiently handled.

99. As per claim 64, Banks teaches the invention as claimed, including the computer system as claimed in claim 58, wherein the application server further performs functions of:

monitoring an amount of time to return a result by the computer system (col. 5 lines 44-47; col. 6 lines 66-67);

determining whether the amount of time has exceeded an upper limit (col. 5 lines 54-56); and

if the amount of time has exceeded the upper limit but has not dropped below a lower limit, not processing new work items (col. 5 lines 54-56).

100. As per claim 70, Banks teaches the invention as claimed, including the computer-readable medium as claimed in claim 65, wherein the method further comprises:

monitoring a quantity of work being performed by the computer system (col. 5 lines 44-47);

determining whether the quantity has exceeded an upper limit (col. 5 lines 54-56); and

if the quantity has exceeded the upper limit but has not dropped below a lower limit, not accepting new requests into the computer system (col. 5 lines 54-56).

101. As per claim 71, Banks teaches the invention as claimed, including the computer-readable medium as claimed in claim 65, wherein the method further comprises:

monitoring an amount of time to return a result by the computer system (col. 5 lines 44-47; col. 6 lines 66-67);

determining whether the amount of time has exceeded an upper limit (col. 5 lines 54-56); and

if the amount of time has exceeded the upper limit but has not dropped below a lower limit, not processing new work items (col. 5 lines 54-56).

Response to Arguments

102. **Applicant's arguments filed February 17, 2005 have been fully considered but they are not persuasive.**

103. Applicant argues that the Shah reference is deficient because it allegedly "*makes no mention of a pool of generic threads as defined by the claims.*" Applicant adds, "*While the*

listening threads are identical, they are not generic threads, nor is there any mention of a pool of generic threads as defined by claim 1."

104. Applicant's suggestion that the threads of Shah are not "generic" threads is based on the idea that Shah defines threads for various functions, i.e. "client", "client listening", "server", and "server listening" threads. Each of these performs different functions, though the listening threads have essentially the same code (col. 3 lines 57-61). However, it should be noted that the function a thread performs, which is based on its programmed source code (see, e.g., col. 4 lines 1-24), is distinct from a thread's structure in the overall system. That is, multithreaded operating systems generally have many threads of execution. Threads may be created to carry out particular functions, and once the thread has completed its function, it may be destroyed. Another model of multithreading, such as the one used by Shah, utilizes thread pools, where worker threads and dispatch threads are used to handle requests. However, Shah notes that such a model typically involves a great deal of overhead as context switches occur between the various threads that process the requests (col. 1 lines 46-61). Thus, Shah minimizes the context switches by providing an interprocess communication messaging system. Though Shah has specialized threads for listening for requests and replies, the underlying structure of the threads are identical.

For example, the multithreaded Windows NT operating system may have a pool of worker threads. When the system starts, one of the threads is invoked as a server listening thread, while another is invoked as a server thread. The threads continue to execute these functions. Though the threads perform distinct operations, they are all pulled from the same pool of generic threads and subsequently given their specialized function.

105. Applicant argues that the Banks reference is deficient because there is allegedly "*no mention of a lower limit in Banks.*" Applicant adds, "*Determining if a queue has dropped below a maximum length is not the same as determining if a queue has dropped below a lower limit.*"

106. Examiner notes the differences that Applicant is attempting to point out between Banks and the claimed invention. However, these differences are not apparent in the claim language. Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993). It appears as though Applicant intends the claim to indicate that in response to an overload condition, incoming requests are rejected until the overload condition is resolved, wherein the resolution of the overload condition is indicated by the amount of work in the work queue dropping below a threshold that indicates that there is room in the queue for several additional work items. On the other hand, Applicants' specification actually indicates that the lower limit may be equal to the upper limit (pg. 44, line 6). If this embodiment is chosen, there is absolutely no distinction between Banks and the claimed invention. Particularly, when a queue is overloaded (exceeded an upper limit), Banks rejects new requests until the queue drops below the overload threshold (dropped below a lower limit).

107. The remainder of Applicant's arguments are related to claims being dependent on or similar to claims that have been argued above.

Conclusion

108. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Syed J. Ali whose telephone number is (571) 272-3769. The examiner can normally be reached on Mon-Fri 8-5:30, 2nd Friday off.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Meng-Ai T. An can be reached on (571) 272-3756. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Syed Ali
May 9, 2005

MENG-AL T. AN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100